

BAB IV

PROSES PEMBUATAN ALAT

4.1 Pembuatan Perangkat Keras

Pembuatan perangkat keras pada perancangan alat ini dengan tahapan realisasi desain sitem menjadi sebuah prototipe tong sampah pintar yang mampu melakukan klasifikasi sampah logam dan nonlogam secara otomatis. Pada tahap ini dilakukan proses perakitan seluruh komponen sesuai dengan rancangan yang telah dibuat, meliputi sistem kendali, sensor, actuator, catu daya berbasis energi surya, dan mekanisme pemilah sampah. Tujuan dari tahap ini adalah menghasilkan sistem yang mampu melakukan deteksi, klalsifikasi, dan pemilahan sampah logam maupun nonlogam secara otomatis serta melakukan kondisi energi secara *real-time*.

Tahap pembuatan perangkat keras diawali dengan menyiapkan seluruh komponen yang akan digunakan, dan dilakukan proses perakitan dengan diagram rangkaian dengan menggunakan aplikasi Fritzing. Perancangan diagram rangkaian dilakukan secara bertahap mulai dari sistem catu daya, pemasangan mikrokontroler, sensor, aktuator, hingga modul komunikasi agar seluruh komponen dapat terhubung menjadi satu sistem yang terintegrasi.

4.1.1 Pembuatan Alat

1. Sistem Catu Daya

Dengan menggunakan panel surya 5V 10W sebagai sumber energi utama. Energi yang dihasilkan panel surya disalurkan menuju Solar Charge Controller (CN3791). Untuk proses pengisian penyimpanan energi pada Baterai Li-ion 18650 sebesar 2750 mAh terdiri atas buah. Berdasarkan perhitungan **persamaan 2.3** maka energi yang dihasilkan oleh panel surya untuk pengisian selama 9 jam yaitu sebesar 108 Wh. Maka energi efektif pada panel tersebut berdasarkan **persamaan 2.4** dengan asumsi 85% adalah 43,24 Wh. Maka nilai Daya rata rata berdasarkan perhitungan **persamaan 2.6** adalah 1,91 W.

Tegangan keluar baterai selanjutnya diatur menggunakan DC Step-Up Converter (MT3608) sehingga menghasilkan tegangan 5 Volt yang stabil untuk menyuplai ESP32 DevKit V1, sensor, motor servo, LCD, Sensor INA219, dan seluruh komponen sistem. Berdasarkan **persamaan 2.10** maka nilai kapasitas baterai sebesar :

$$C_{total} = 5 \times 2750 = 13750 \text{ mAh} = 13,75 \text{ Ah}$$

Berdasarkan persamaan **2.11** dapat diperhitungkan energi nominal maka energi :

$$\begin{aligned} E_{nominal} &= 3,7 \times 13,75 \text{ Ah} \\ &= 50,875 \text{ Wh} \end{aligned}$$

Energi tersebut efektif dengan menggunakan efisiensi 85% menggunakan **persamaan 2.12** :

$$\begin{aligned} E_{efektif} &= 50,875 \times 0,85 \\ &= 43,24 \end{aligned}$$

Berdasarkan perhitungan tersebut maka kapasitas tersebut dapat memenuhi kebutuhan energi sistem selama 24 jam.

2. Pemasangan ESP32 DevKit V1

Selanjutnya dilakukan pemasangan mikrokontroler ESP32 DevKit V1 dengan Pin 38 sebagai Pusat pengedali sistem. ESP32 dihubungkan dengan sensor proximity induktif LJ12A3-4-Z/BY (PNP) untuk mendeteksi sampah logam, sensor proximity kapasitif LJC18A3-H-Z/BY untuk mendeteksi sampah nonlogam, serta dua sensor ultrasonik HC-SR04 yang berfungsi mengukur tingkat kepenuhan tong sampah logam dan nonlogam.

3. Aktuator

Aktuator dipasang dua motor servo MG996R yang berfungsi sebagai mekanisme pengarah sampah menuju tong sesuai dengan hasil

klasifikasi. Selain itu, LCD 16×2 berbasis I2C dipasang sebagai media untuk menampilkan informasi hasil pembacaan sensor, status sistem, serta data monitoring energi.

4. Pemantauan Kondisi Sistem Catu Daya

Untuk memantau kondisi sistem catu daya, digunakan sensor INA219 sebagai sensor pengukuran tegangan dan arus DC. Sensor INA219 dihubungkan dengan ESP32 melalui komunikasi Inter-Integrated Circuit (I2C) menggunakan jalur SDA dan SCL, sehingga proses pembacaan data dapat dilakukan secara langsung tanpa memerlukan modul konverter komunikasi tambahan. Data tegangan dan arus yang diperoleh digunakan untuk memantau kondisi catu daya, sedangkan nilai daya listrik dapat dihitung berdasarkan hasil pengukuran tersebut untuk mendukung proses monitoring kinerja sistem secara keseluruhan.

Setelah seluruh komponen selesai dirakit, dilakukan pemeriksaan terhadap seluruh koneksi kabel untuk memastikan tidak terjadi kesalahan pemasangan maupun hubungan singkat (*short circuit*). Selanjutnya dilakukan pengujian pada setiap komponen secara individual untuk memastikan sensor, aktuator, sistem catu daya, komunikasi data, serta tampilan LCD telah berfungsi sesuai dengan perancangan sebelum dilakukan pengujian sistem secara keseluruhan.

4.1.2 Alat dan Bahan

Dalam persiapan alat dan bahan yang akan digunakan untuk perancangan sistem yang telah dibuat. Berikut daftar bahan dapat dilihat pada **tabel 4.1**

Tabel 4. 1 Alat dan Bahan Pendukung

No	Nama Bahan	Jumlah
1.	ESP32 DevKit V1	1 buah
2.	Baterai Li-ion 18650/2750 mAh	8 buah
3.	Kawat Pushrod Servo	3 buah
4.	Baud Servo	3 buah
5.	Cetak Support Servo 3D	2 buah
6.	Kayu Reng	5 buah
7.	Triplek	2 lembar

No	Nama Bahan	Jumlah
8.	Paku	1 plastik
9.	Penyangga Siku 1 + Baud	13 pcs
10.	Kabel AWG 24	3 meter
11.	Penyangga Siku 2 + Baud	10 pcs
12.	Komponen PCB	
13.	LJ12A3-4-Z BY PNP Induktif Proximity Sensor	1 buah
14.	SR04 Ultrasonic Sensor	2 buah
15.	MT3608 DC Step Up Converter	1 buah
16.	Panel Surya Anbolt 5V 10W	1 buah
17.	CN3791 Solar Charger Controller	1 buah
18.	sensor proximity kapasitif LJC18A3-H-Z/	1 buah
19.	Micro Servo MG90S	2 buah
20.	LCD i2c	1 buah
21.	Breadboard	1 buah
22.	Jumper male-female, male-male	1 roll
23.	Resistor 10 k Ω	12 buah
24.	Sensor INA219	1 buah
25.	BOX X9	1 buah
26.	Skrup Tambahan Rangka	32 buah
27.	Tempat Sampah 7 L	2 buah
28.	Step Up XL6019	1 buah

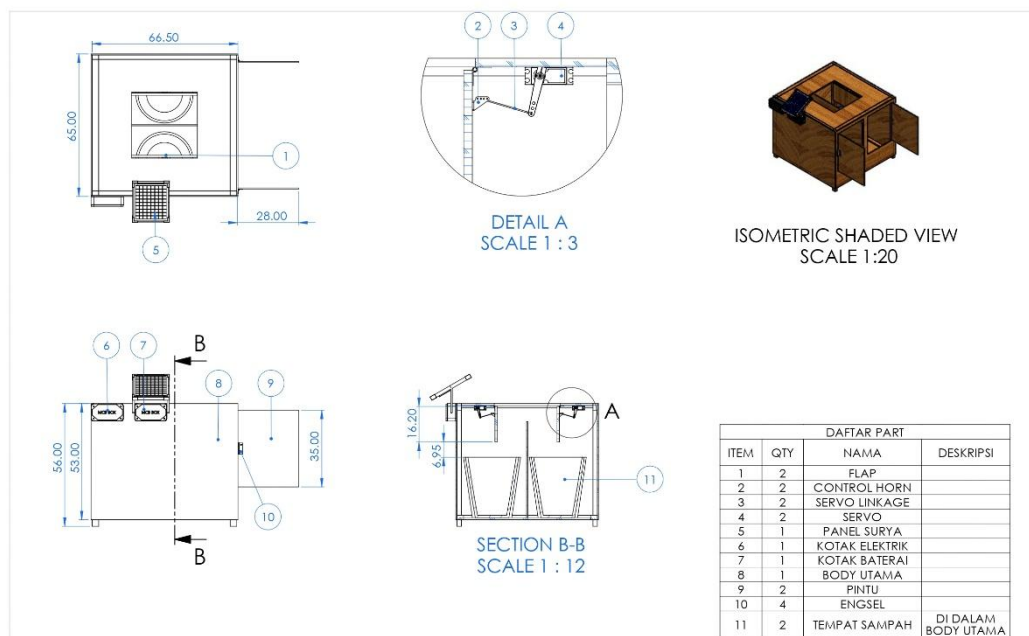
Pada **tabel 4.2** merupakan alat yang digunakan untuk pembuatan perangkat keras (*hardware*) pada alat ini.

Tabel 4. 2 Alat Pendukung

No.	Nama Alat	Jumlah
1.	Multimeter	1 Buah
2.	Obeng +	1 Buah
3.	Obeng -	1 Buah
4.	Tang Potong	1 Buah
5.	Tang Kombinasi	1 Buah
6.	Solder	1 Buah
7.	Bor	1 Buah
8.	Gergaji	1 Buah

4.1.3 Langkah Langkah Pembuatan

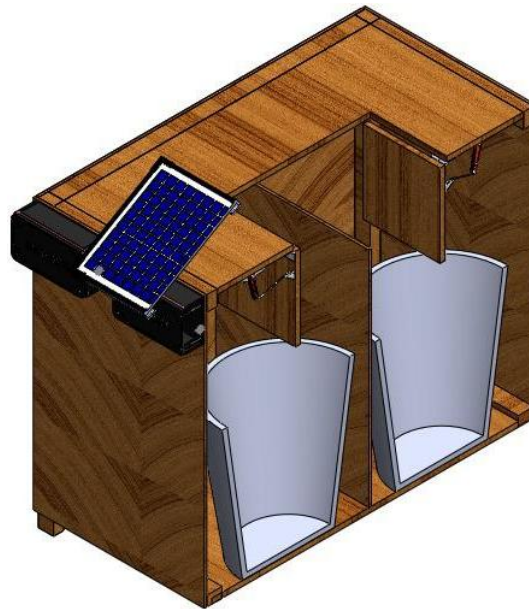
Pada proses pembuatan perangkat keras “Tong Sampah Pintar berbasis energi surya dengan klasifikasi sampah logam dan nonlogam dengan menggunakan multisensor dan ESP32. Dengan menyiapkan bahan dan alat yang tercantum pada tabel diatas. Langkah ini bertujuan untuk memastikan bahwa semua alat dan material telah tersedia dan memenuhi spesifikasi yang dibutuhkan untuk perancangan alat. Setelah alat dan bahan lengkap langkah berikutnya ialah perakitan alat dan bahan.



Gambar 4. 1 Desain 3D

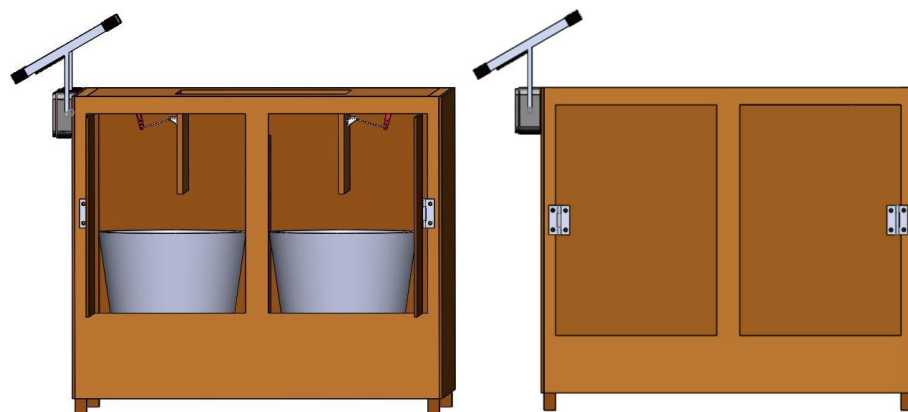
Pada **Gambar 4.1** merupakan Desain 3D untuk proses pembuatan perangkat keras “Tong Sampah Pintar berbasis energi surya dengan klasifikasi sampah logam dan nonlogam dengan menggunakan multisensor dan ESP32. Dengan menyiapkan bahan dan alat yang tercantum pada tabel diatas. Langkah ini bertujuan untuk memastikan bahwa semua alat dan material telah tersedia dan memenuhi spesifikasi yang dibutuhkan untuk perancangan alat. Setelah alat dan bahan lengkap langkah berikutnya ialah perakitan alat dan bahan. Pemasangan

dilakukan mengikuti rancangan visualisasi 3D pada **Gambar 4.1** yang telah dibuat sebelumnya, seperti gambar 3D dibawah ini :



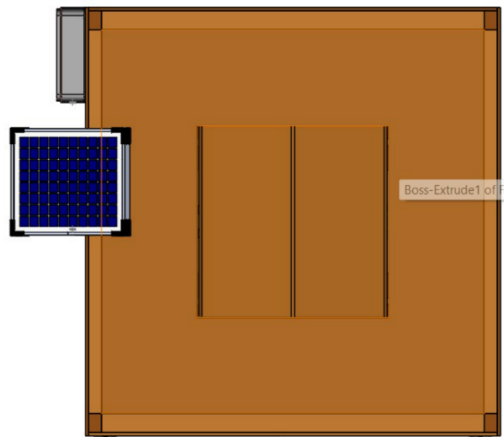
Gambar 4. 2 *Alat Tampak samping*

Pada **Gambar 4.2** ditunjukkan desain 3D body Alat Tong Sampah tampak Samping



Gambar 4. 3 *Alat Tampak Depan*

Pada **Gambar 4.3** ditunjukkan desain 3D body Alat Tong Sampah tampak Depan



Gambar 4. 4 *Alat Tampak Atas*

Pada **Gambar 4.4** ditunjukkan desain 3D body Alat Tong Sampah tampak Atas.

Desain ini merupakan tampilan mounting dari alat ini berbagai sudut, untuk mempermudah penempatan komponen pada perancangan alat ini. Tinggi dari body tempat sampah adalah 53 cm. Dengan Tinggi efektif 50 cm, dan tinggi dari Tempat Sampah logam dan nonlogam tersebut 27 cm. Dengan jarak flap pemisah sampah dengan tempat sampah sekitar 3,8 cm. untuk pemasangan komponen alat Mikrokontroler, sensor, baterai, dan servo terdapat pada Box X9 **Gambar 4.5**.



Gambar 4. 5 *Box Komponen X9*

Realisasi desain : Pembuatan Body Tempat Sampah

dalam merealisasikan desain pembuatan body sampah ini realisasi desain 3D menjadi bentuk fisik. Body sampah dibuat dengan menggunakan material triplek yang dipotong sesuai dengan ukuran dan bentuk yang telah ditentukan pada rancangan menggunakan *CNC (Computer Numerical Control) Machine*. Dengan pemotongan menggunakan *CNC Machine* bertujuan memperoleh hasil pemotongan yang lebih presisi dan sesuai dengan bentuk rancangan, terutama pada bagian bagian yang memiliki dan pola pemotongan tertentu.



Gambar 4. 6 Pemotongan Triplek menggunakan CNC

Proses pemotongan material triplek menggunakan *CNC Machine* ditunjukkan pada **Gambar 4.6** pemotongan dilakukan berdasarkan pola yang telah dirancang sehingga setiap bagian body memiliki ukuran dan pola pemotongan yang sesuai dengan rancangan alat. Proses ini menghasilkan struktur body utama tempat sampah yang selanjutnya digunakan sebagai tempat pemasangan wadah sampah, mekanisme pemilah, sensor, servo, dan box elektronik.

Setelah proses pemotongan menggunakan *CNC Machine* sudah selesai, bagian bagian triplek kemudian dirakit menjadi body utama tempat sampah. Proses perakitan dilakukan dengan menyusun bagian sisi, bagian belakang, bagian atas, dan pintu depan sesuai dengan rancangan yang telah dibuat. Setiap bagian disambungkan menggunakan engsel dan pengikat sehingga membentuk struktur body yang kokoh. Pada bagian dalam body juga disediakan ruang untuk penempatan wadah sampah, sedangkan bagian depan dibuat menjadi dua pintu yang dapat dibuka untuk memudahkan proses pemasangan dan pengambilan wadah sampah. Hasil pemasangan body triplek ditunjukkan pada **Gambar 4.7**



Gambar 4. 7 Perakitan Body Alat

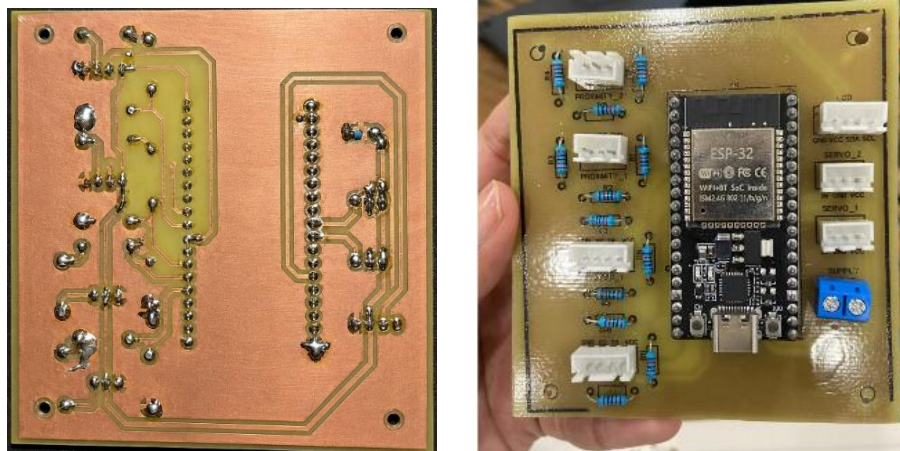


Gambar 4. 8 Pengecetan Body Alat

Pada **Gambar 4.8** Setelah perakitan body telah dilakukan, kemudian dilakukan pengecetan anti jamur dan anti rayap bertujuan memperpanjang kualitas pemakaian body tempat sampah dengan material triplek. Dan dilakukan pengecekan akhir pada alat untuk memastikan perangkat keras berfungsi secara optimal.

Hasil Akhir Alat

Setiap komponen dalam PCB disesuaikan dan dirancang sesuai layout perancangan untuk menjamin fungsi indikator berjalan sesuai dengan fungsi. Proses penyolderan dilakukan untuk penempatan mikrokontroler, sensor, dan servo.



Gambar 4. 9 *Desain PCB*

Pada **Gambar 4.9** merupakan Perakitan sistem dilakukan dengan menyolder seluruh komponen pada papan PCB sesuai dengan diagram wiring dan pin mapping yang telah dirancang. ESP32 DevKit V1 dipasang sebagai mikrokontroler utama yang berfungsi mengendalikan seluruh proses pada sistem. Sensor induktif dihubungkan ke pin GPIO34 untuk mendeteksi sampah logam, sedangkan sensor kapasitif dihubungkan ke GPIO35 untuk mendeteksi sampah nonlogam. Dua sensor ultrasonik HC-SR04 dipasang untuk memantau tingkat kepenuhan tempat sampah, dengan sensor pertama menggunakan GPIO25 sebagai pin Trigger dan GPIO26 sebagai pin Echo, sedangkan sensor kedua menggunakan GPIO27 sebagai pin Trigger dan GPIO32 sebagai pin Echo. Motor servo sebagai aktuator pemilah sampah dipasang pada GPIO18 untuk pemilahan sampah logam dan GPIO17 untuk pemilahan sampah nonlogam. Selanjutnya, LCD 16×2 dengan antarmuka I2C dihubungkan ke GPIO21 (SDA) dan GPIO22 (SCL) sebagai media penampil informasi. Sensor INA219 juga dihubungkan pada jalur I2C yang sama, yaitu GPIO4 sebagai SDA dan GPIO5 sebagai SCL, untuk melakukan pemantauan tegangan dan arus pada sistem catu daya. Seluruh komponen memperoleh suplai daya dari sistem energi surya yang terdiri atas panel surya, solar charge controller, baterai Li-ion 18650, dan modul DC Step-Up MT3608 Step Up XL609. Setelah proses penyolderan selesai, dilakukan pemeriksaan terhadap setiap jalur PCB dan koneksi antar komponen untuk memastikan tidak terdapat kesalahan pengkabelan maupun hubungan singkat (short circuit). Tahap ini bertujuan agar seluruh

komponen dapat berkomunikasi dengan baik dan sistem dapat beroperasi sesuai dengan rancangan.

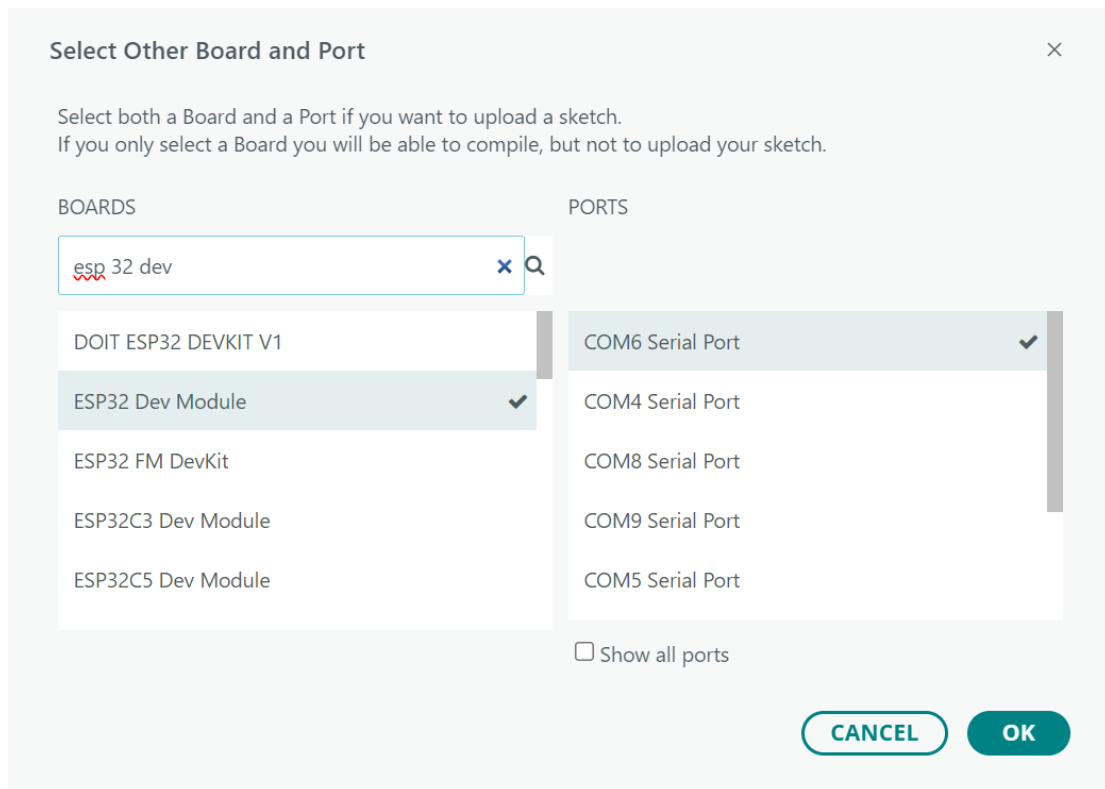
4.2 Pembuatan Perangkat Lunak (Software)

Pembuatan perangkat Lunak atau *Software* dilakukan dengan membuat pemrograman sistem, pemograman sistem alat ini dibuat menggunakan aplikasi Arduino IDE software yang mendukung untuk mikrokontroler ESP32 DevKit V1. Perangkat lunak ini berfungsi sebagai pengendali utama yang mengatur seluruh proses kerja sistem. Pada tahap awal program, dilakukan inisialisasi seluruh komponen yang digunakan, meliputi sensor induktif, sensor kapasitif, dua sensor ultrasonik HC-SR04, sensor INA219, motor servo, LCD I2C, serta modul komunikasi Wi-Fi pada ESP32. Setelah proses inisialisasi selesai, ESP32 melakukan koneksi ke jaringan Wi-Fi dan server Blynk agar data hasil pembacaan sensor dapat dipantau secara real-time.

4.2.1 Pembuatan Program Arduino IDE

Pembuatan program diawali dengan inisialisasi komponen yang digunakan pada alat Rancang Bangun Tong Sampah Pintar, meliputi sensor induktif, sensor kapasitif, sensor ultrasonik HC-SR04, sensor INA219, motor servo, LCD I2C, serta konfigurasi koneksi Wi-Fi dan Blynk. Kemudian dengan melakukan pemilihan

jenis Board, yaitu ESP32 Dev Module dan pemilihan Port Komunikasi, yaitu COM6 Serial Port. Proses ini ditampilkan pada **gambar 4.10** berikut :



Gambar 4. 10 Pemilihan Board dan Port

kemudian melakukan *verify* kode program dan melakukan *upload* jika sudah dan setelah proses verifikasi berhasil tanpa ditemukan kesalahan. Maka kode telah berhasil terkirim ke perangkat dan siap untuk dijalankan. Proses pengunggahan dapat dilihat pada Langkah Langkah dibawah ini.

Langkah Langkah pembuatan sketch pada Arduino IDE sebagai berikut :

Langkah 1 : Konfigurasi Blynk (#define)

Pada konfigurasi awal program dengan mendefinisikan parameter yang diperlukan untuk menghubungkan ke mikrokontroler yaitu ESP32 DevKit V1 dengan platfrom Blynk IoT. Parameter tersebut meliputi Template ID, Template Name, dan Authentication Token yang diperoleh dari akun Blynk. Ketiga parameter didefinisikan menggunakan direktif `#define` sehingga dapat digunakan sebagai

identitas perangkat pada saat proses autentikasi dengan server Blynk. Pada **Gambar 4.11** merupakan gambar pemanggilan platform Blynk

```
1  #define BLYNK_TEMPLATE_ID "TMPL6dJpvCjIN"
2  #define BLYNK_TEMPLATE_NAME "Metal Shorter By Anandya"
3  #define BLYNK_AUTH_TOKEN "_fPmNJWcrn4S7CZVzqPaAd8i55n20uv0"
```

Gambar 4. 11 Pemanggilan Platform BYLNK

BLYNK_TEMPLATE_ID merupakan identitas template proyek yang dibuat pada platform Blynk, **BLYNK_TEMPLATE_NAME** berisi nama template yang digunakan sebagai identitas proyek, sedangkan **BLYNK_AUTH_TOKEN** merupakan token autentikasi yang berfungsi sebagai kunci akses agar ESP32 dapat terhubung dengan server Blynk. Setelah konfigurasi tersebut dilakukan, ESP32 dapat mengirimkan data hasil pembacaan sensor ke dashboard Blynk melalui jaringan Wi-Fi untuk keperluan monitoring secara real-time.

Langkah 2 : Pemanggilan Library (#include)

pada Langkah ini dilakukan pemanggilan library dengan menggunakan direktif `#include`. Pada bagian ini memiliki fungsi untuk mendukung proses pemrograman sehingga ESP32 dapat berkomunikasi dengan sensor, aktuator, dan perangkat lainnya. Pada bagian awal program dilakukan pemanggilan library pada **Gambar 4.12** sebelum proses deklarasi variable dan penyusunan fungsi utama.

```
5  #include <WiFi.h>
6  #include <BlynkSimpleEsp32.h>
7  #include <Wire.h>
8  #include <Adafruit_INA219.h>
9  #include <LiquidCrystal_I2C.h>
10 #include <ESP32Servo.h>
11 #include <Preferences.h>
```

Gambar 4. 12 Library perangkat

library **WiFi.h** digunakan untuk menghubungkan ESP32 dengan jaringan Wi-Fi sehingga perangkat dapat mengakses internet. Library **BlynkSimpleEsp32.h** berfungsi sebagai antarmuka komunikasi antara ESP32 dan platform Blynk IoT untuk mengirimkan data hasil pembacaan sensor serta menerima perintah dari aplikasi Blynk secara real-time. Library **Wire.h** digunakan untuk mengatur komunikasi I2C yang dimanfaatkan oleh LCD 16×2 dan sensor INA219 melalui jalur SDA dan SCL.

Selanjutnya, library **Adafruit_INA219.h** digunakan untuk membaca data tegangan dan arus dari sensor INA219. Data tersebut digunakan untuk memantau kondisi catu daya pada sistem berbasis energi surya. Library **LiquidCrystal_I2C.h** berfungsi untuk mengendalikan tampilan LCD 16×2 berbasis I2C sebagai media penyampaian informasi kepada pengguna, seperti jenis sampah, tingkat kepenuhan tempat sampah, serta hasil pengukuran sensor. Selain itu, library **ESP32Servo.h** digunakan untuk mengendalikan pergerakan motor servo yang berfungsi sebagai aktuator dalam proses pemilahan sampah logam dan nonlogam. Dengan memanggil seluruh library tersebut, ESP32 dapat mengintegrasikan seluruh komponen perangkat keras sehingga sistem dapat bekerja sesuai dengan perancangan. Library **Preferences.h** digunakan untuk menyediakan fungsi penyimpanan dan pembacaan data pada memori non-volatile ESP32. Penggunaan library ini memungkinkan data atau parameter tertentu pada sistem tetap tersimpan meskipun mikrokontroler mengalami proses *restart* atau kehilangan catu daya.

Langkah 3 : konfigurasi parameter sistem dan deklarasi konstanta

Pada tahap ini dengan melakukan konfigurasi parameter yang digunakan dalam program, dengan mengkonfigurasi pengaturan jaringan Wi-Fi, komunikasi I2C, aktivasi sensor ultrasonik, serta menentukan nilai ambang batas (threshold) sensor. Parameter didefinisikan bagian awal program agar mudah diubah sesuai kebutuhan tanpa mempengaruhi logika utama sistem. Konfigurasi ini berfungsi sebagai dasar bagi ESP32 DevKit V1 dalam menjalankan proses komunikasi dan pembacaan data dari setiap sensor yang digunakan pada **Gambar 4.13**

```

13 // GANTI WIFI DI SINI
14 char ssid[] = "Anan";
15 char pass[] = "Anandya1809";
16 //— I2C —————
17 #define I2C_SDA 4
18 #define I2C_SCL 5
19
20 // — FITUR ULTRASONIK —————
21 #define ULTRASONIK_AKTIF true
22
23
24 #define THRESHOLD_INDUKTIF 600
25 #define THRESHOLD_KAPASITIF 1000
26

```

Gambar 4. 13 konfigurasi parameter sistem

variabel ssid dan pass digunakan untuk menyimpan nama jaringan (**SSID**) dan kata sandi (**password**) Wi-Fi yang akan digunakan oleh ESP32 untuk terhubung ke internet. Selanjutnya, direktif `#define I2C_SDA` dan `#define I2C_SCL` digunakan untuk menentukan pin komunikasi I2C yang digunakan oleh LCD I2C dan sensor INA219. Parameter `ULTRASONIK_AKTIF` digunakan untuk mengaktifkan fungsi pembacaan sensor ultrasonik, sedangkan `THRESHOLD_INDUKTIF` dan `THRESHOLD_KAPASITIF` digunakan sebagai nilai ambang batas untuk menentukan kondisi deteksi sensor induktif dan sensor kapasitif. Nilai ambang batas tersebut menjadi acuan bagi ESP32 dalam proses klasifikasi sampah logam dan nonlogam.

Langkah 4 : Deklarasi Pin dan Konfigurasi Perangkat

Sebelum ESP32 dapat berkomunikasi dengan seluruh komponen perangkat keras, diperlukan proses deklarasi pin input dan output yang digunakan oleh masing-masing sensor dan aktuator. Pada tahap ini juga dilakukan konfigurasi parameter motor servo, seperti sudut buka, sudut tutup, dan waktu tunda (delay) pergerakan servo.

```

● 27 // — PIN DEFINITIONS
28 #define PIN_INDUKTIF      34
29 #define PIN_KAPASITIF    35
30 #define PIN_SERVO_A      18
31 #define PIN_SERVO_B      17
32 #define TRIG1            25
33 #define ECHO1            26
34 #define TRIG2            27
35 #define ECHO2            32
36
37 // — SERVO CONFIG
38 #define SERVO_A_TUTUP     90
39 #define SERVO_A_BUKA     180
40 #define SERVO_B_TUTUP     10
41 #define SERVO_B_BUKA     90
42 #define SERVO_OPEN_MS    2500UL

```

Gambar 4. 14 Deklarasi Pin dan konfigurasi Perangkat

pendefinisian pin pada **Gambar 4.14** yang digunakan oleh setiap komponen pada sistem. Sensor induktif dihubungkan ke GPIO 34, sensor kapasitif ke GPIO 35, motor Servo A ke GPIO 18, dan Servo B ke GPIO 19. Selanjutnya, sensor ultrasonik pertama menggunakan GPIO 25 sebagai pin Trigger dan GPIO 26 sebagai pin Echo, sedangkan sensor ultrasonik kedua menggunakan GPIO 27 sebagai pin Trigger dan GPIO 32 sebagai pin Echo. Selain deklarasi pin, pada bagian ini juga ditentukan parameter sudut buka dan tutup motor servo serta waktu tunda pergerakan servo.

Langkah 5 : Konfigurasi Parameter Sensor Ultrasonik dan Blynk

Konfigurasi beberapa parameter sistem yang digunakan sebagai acuan dalam proses pembacaan sensor ultrasonik, penentuan tingkat kepenuhan tempat sampah, pencegahan pembacaan berulang (*debounce*), serta pengaturan interval pengiriman data ke aplikasi Blynk. Parameter pada **Gambar 4.15** konfigurasi

parameter sensor HCSR-04 ditentukan diawal program sehingga memudahkan proses penyesuaian sistem tanpa mengubah algoritma utama.

```

45 //—·ULTRASONIK·CONFIG·—————
46 #define JARAK_PENUH·····5····
47 #define JARAK_KOSONG·····24···
48 #define WARN_PERSEN·····20···
49
50 //—·BLYNK·UPDATE·INTERVAL·—————
51 #define BLYNK_UPDATE_MS···10000

```

Gambar 4. 15 Konfigurasi Parameter sensor ultrasonik dan Blynk

parameter JARAK_PENUH ditetapkan sebesar **5 cm** sebagai batas jarak yang menunjukkan bahwa tempat sampah telah mencapai kondisi penuh. Parameter JARAK_KOSONG sebesar **30 cm** digunakan sebagai acuan ketika tempat sampah berada dalam kondisi kosong. Selanjutnya, WARN_PERSEN ditetapkan sebesar **20%** sebagai batas peringatan ketika kapasitas tempat sampah hampir penuh. Selain itu, parameter DEBOUNCE_MS sebesar **10000 ms** digunakan untuk memberikan jeda waktu pembacaan sensor sehingga dapat mengurangi kesalahan deteksi akibat pembacaan yang terlalu cepat. Adapun BLYNK_UPDATE_MS sebesar **10000 ms** digunakan untuk mengatur interval pengiriman data hasil pembacaan sensor ke platform Blynk sehingga proses monitoring dapat dilakukan secara berkala dan real-time.

Langkah 6 : Pembuatan Objek dan Deklarasi Variabel Global

Pada tahap ini dengan melakukan pembuatan objek dari setiap library yang digunakan serta mendeklarasikan variable global. Objek dibuat agar ESP32 dapat mengakses fungsi yang disediakan setiap library, sedangkan variable global digunakan untuk menyimpan data dan status sistem yang akan digunakan secara berulang selama program dijalankan.

```

54 // — OBJEK —————
55 Adafruit_INA219 ina219;
56 LiquidCrystal_I2C lcd(0x27, 16, 2);
57 Servo servoA;
58 Servo servoB;
59 BlynkTimer timer;
60
61 // — COUNTER —————
62 int counterLogam = 0;
63 int counterNonLogam = 0;
64
65 // — STATE TRACKING —————
66 bool prevLogam = false;
67 bool prevBenda = false;

```

Gambar 4. 16 Pembuatan Objek dan Deklarasi Variabel Global

Pada **Gambar 4.16** merupakan objek **ina219** dibuat dari library **Adafruit_INA219** untuk mengakses fungsi pembacaan tegangan dan arus pada sensor INA219. Objek **lcd** digunakan untuk mengendalikan tampilan LCD 16×2 berbasis I2C dengan alamat perangkat **0x27**. Selanjutnya, objek **servoA** dan **servoB** dibuat dari library **ESP32Servo** untuk mengendalikan pergerakan motor servo yang berfungsi sebagai aktuator pemilah sampah logam dan nonlogam. Selain itu, objek **timer** dari library **BlynkTimer** digunakan untuk mengatur proses eksekusi fungsi secara periodik, seperti pengiriman data ke aplikasi Blynk.

Pada **Tabel 4.3** merupakan bagian berikutnya dilakukan deklarasi beberapa variabel global yang digunakan selama program berjalan. Variabel **counterLogam** dan **counterNonLogam** digunakan untuk menghitung jumlah sampah logam dan nonlogam yang berhasil diklasifikasikan. Sementara itu, variabel bertipe **boolean** yaitu **prevLogam** dan **prevBenda** digunakan untuk menyimpan status pembacaan sensor sebelumnya (*state tracking*), sehingga ESP32 dapat membedakan kondisi sebelum dan sesudah suatu objek terdeteksi serta menghindari proses klasifikasi yang terjadi secara berulang.

Tabel 4. 3 Deklarasi Variabel Global

Objek/Variabel	Tipe	Fungsi
INA219	Adafruit_INA219	Membaca tegangan dan arus sensor INA219.
LCD	LiquidCrystal_I2C	Mengendalikan tampilan LCD 16×2.
Servo A	Servo	Mengendalikan servo pemilah sampah logam.
Servo B	Servo	Mengendalikan servo pemilah sampah nonlogam.
Timer	BlynkTimer	Mengatur eksekusi fungsi secara periodik
counterLogam	Int	Menghitung jumlah sampah logam.
counterNonLogam	Int	Menghitung jumlah sampah nonlogam.
prevLogam	bool	Menyimpan status deteksi sensor logam sebelumnya.
prevBenda	bool	Menyimpan status keberadaan objek sebelumnya.

Langkah 7 : Konfigurasi Pewaktuan Sistem (Timing)

Pada tahap ini dilakukan konfigurasi parameter waktu (*timing*) yang digunakan untuk mengatur interval pembacaan sensor dan pembaruan tampilan LCD. Variabel bertipe unsigned long digunakan untuk menyimpan waktu terakhir proses tertentu dijalankan berdasarkan fungsi millis(), sehingga program dapat menjalankan beberapa proses secara periodik tanpa menggunakan fungsi delay(). Selain itu, direktif #define LCD_REFRESH_MS digunakan untuk menentukan interval pembaruan tampilan LCD setiap 500 ms, sehingga informasi yang ditampilkan tetap diperbarui secara berkala tanpa mengganggu proses pembacaan sensor maupun komunikasi data.

```

74 // — TIMING —————
75 unsigned long lastDeteksi    = 0;
76 unsigned long lastLCDUpdate = 0;
77 #define LCD_REFRESH_MS      500
78 #define DEBOUNCE_MS         800

```

Gambar 4. 17 Konfigurasi Timing sistem

Berdasarkan **gambar 4.17** , pada sketch ini tersebut terdiri dari tiga bagian utama, yaitu:

1. **lastDeteksi** digunakan untuk menyimpan waktu terakhir sistem melakukan proses deteksi sampah.
2. **lastLCDUpdate** digunakan untuk menyimpan waktu terakhir pembaruan tampilan pada LCD.
3. **LCD_REFRESH_MS** merupakan konstanta yang menentukan interval pembaruan LCD, yaitu setiap 500 milidetik. Dengan pengaturan ini, tampilan LCD dapat diperbarui secara berkala tanpa menghentikan proses lain yang sedang berjalan sehingga sistem tetap responsif.

Langkah 8 : Implementasi Fungsi Pembacaan Sensor Analog

Pada tahap ini dilakukan pembuatan fungsi pembacaan sensor analog yang terdiri atas sensor induktif dan kapasitif. Kedua sensor digunakan dalam proses klasifikasi sampah berdasarkan karakteristik material yang terdeteksi. Nilai keluaran masing-masing sensor dibaca menggunakan fungsi `analogRead()`, kemudian dibandingkan dengan nilai *threshold* yang telah ditentukan. Hasil pembacaan tersebut selanjutnya digunakan sebagai dasar pengambilan keputusan sistem dalam menentukan jenis sampah yang terdeteksi dan mengendalikan proses pemilahan.

```

81 // SENSOR ANALOG
82 // =====
83 ✓ bool bacaInduktif() {
84     int val = analogRead(PIN_INDUKTIF);
85     Serial.print("ADC Induktif: ");
86     Serial.print(val);
87     Serial.print(" | ");
88     return (val > THRESHOLD_INDUKTIF);
89 }
90
91 ✓ bool bacaKapasitif() {
92     int val = analogRead(PIN_KAPASITIF);
93     Serial.print("ADC Kapasitif: ");
94     Serial.print(val);
95     Serial.print(" | ");
96     return (val > THRESHOLD_KAPASITIF);
97 }

```

Gambar 4. 18 Implementasi Pembacaan Sensor Analog

Berdasarkan program pada **Gambar 4.18** tersebut, fungsi `bacaInduktif()` digunakan untuk membaca nilai analog dari sensor induktif melalui `PIN_INDUKTIF`. Nilai hasil pembacaan disimpan pada variabel `val`, kemudian ditampilkan pada Serial Monitor dengan keterangan “ADC Induktif”. Sistem selanjutnya membandingkan nilai tersebut dengan `THRESHOLD_INDUKTIF`. Apabila nilai pembacaan lebih besar dari nilai *threshold*, fungsi mengembalikan nilai `true` yang menunjukkan kondisi sensor induktif terdeteksi. Sementara itu, fungsi `bacaKapasitif()` digunakan untuk membaca nilai analog dari sensor kapasitif melalui `PIN_KAPASITIF`. Nilai pembacaan dibandingkan dengan `THRESHOLD_KAPASITIF` dan menghasilkan nilai `true` apabila nilai sensor melebihi batas yang telah ditentukan. Hasil dari kedua fungsi tersebut kemudian digunakan oleh program utama dalam menentukan proses klasifikasi sampah.

Langkah 9 : Pemrograman Sensor Ultrasonik dan Perhitungan Persentase Kepenuhan

Pada tahap ini dilakukan pemrograman sensor ultrasonik HC-SR04 yang digunakan untuk mengukur jarak antara sensor dengan permukaan sampah di dalam tempat sampah. Program mengatur proses pengiriman sinyal melalui pin *trigger* dan menerima sinyal pantulan melalui pin *echo*. Waktu tempuh sinyal ultrasonik kemudian digunakan untuk menentukan jarak permukaan sampah dari sensor.

Selain pengukuran jarak, program juga mengolah nilai jarak tersebut menjadi persentase tingkat kepenuhan tempat sampah. Nilai jarak dibandingkan dengan parameter JARAK_PENUH dan JARAK_KOSONG yang telah ditentukan sebelumnya. Apabila jarak yang terukur lebih kecil atau sama dengan batas jarak penuh, maka persentase kepenuhan ditetapkan sebesar 100%. Sementara itu, apabila jarak berada di antara kondisi penuh dan kosong, program menghitung persentase menggunakan fungsi `map()`. Hasil perhitungan kemudian ditampilkan melalui *Serial Monitor* dan digunakan sebagai informasi tingkat kepenuhan tempat sampah.

```

// =====
// ULTRASONIK
// =====

102 long bacaJarak(int trigPin, int echoPin) {
103     if (!ULTRASONIK_AKTIF) return 999;
104     digitalWrite(trigPin, LOW);
105     delayMicroseconds(2);
106     digitalWrite(trigPin, HIGH);
107     delayMicroseconds(10);
108     digitalWrite(trigPin, LOW);
109     long durasi = pulseIn(echoPin, HIGH, 30000);
110     if (durasi == 0) return 999;
111     return durasi * 0.034 / 2;
112 }
113
114 int hitungPersen(int trigPin, int echoPin) {
115     if (!ULTRASONIK_AKTIF) return 0;
116     long jarak = bacaJarak(trigPin, echoPin);
117     if (jarak >= JARAK_KOSONG) return 0;
118     if (jarak <= JARAK_PENUH) return 100;
119     int persen = map(jarak, JARAK_PENUH, JARAK_KOSONG, 100, 0);
120     Serial.print("Jarak: ");
121     Serial.print(jarak);
122     Serial.print("cm Isi:");
123     Serial.print(persen);
124     Serial.print("% | ");
125     return persen;
126 }
127
128 bool cekBoxPenuh(int trigPin, int echoPin) {
129     return (hitungPersen(trigPin, echoPin) >= 100);
130 }

```

Gambar 4. 19 *Sensor Ultrasonik dan Perhitungan persentase Level Tong*

Berdasarkan program pada **Gambar 4.19** diatas fungsi `bacaJarak()` digunakan untuk memperoleh nilai jarak berdasarkan durasi pantulan gelombang ultrasonik. Selanjutnya, fungsi `hitungPersen()` mengubah nilai jarak tersebut menjadi persentase kepenuhan tempat sampah dengan rentang 0–100%. Nilai persentase tersebut digunakan sebagai salah satu parameter dalam proses pemantauan kondisi tempat sampah secara otomatis.

Langkah 10 : Penyimpanan Data dan Pengendalian Servo

```

132 // =====
133 // SIMPAN DATA
134 // =====
135 void simpanData() {
136     preferences.putInt("logam", counterLogam);
137     preferences.putInt("nonlogam", counterNonLogam);
138 }
139
140 // =====
141 void bukaServoA() {
142     servoA.write(SERVO_A_BUKA);
143     servoA_terbuka = true;
144     servoA_waktuBuka = millis();
145 }
146
147 void bukaServoB() {
148     servoB.write(SERVO_B_BUKA);
149     servoB_terbuka = true;
150     servoB_waktuBuka = millis();
151 }
152
153 void updateServo() {
154     unsigned long sekarang = millis();
155
156     if (servoA_terbuka && (sekarang - servoA_waktuBuka >= SERVO_OPEN_MS)) {
157         servoA.write(SERVO_A_TUTUP);
158         servoA_terbuka = false;
159     }
160
161     if (servoB_terbuka && (sekarang - servoB_waktuBuka >= SERVO_OPEN_MS)) {
162         servoB.write(SERVO_B_TUTUP);
163         servoB_terbuka = false;
164     }
165 }

```

Gambar 4. 20 Pemograman Servo dan Simpan Data

Pada **Gambar 4.20** tahap ini, program mengatur proses penyimpanan data jumlah sampah serta pengendalian mekanisme servo pada alat. Data jumlah sampah logam dan nonlogam disimpan menggunakan library Preferences sehingga data dapat tetap tersimpan pada memori ESP32. Selanjutnya, fungsi bukaServoA() dan bukaServoB() digunakan untuk mengatur servo agar membuka penutup tempat

sampah sesuai jenis sampah yang terdeteksi. Setelah servo terbuka, waktu pembukaan dicatat menggunakan fungsi `millis()`.

Program juga dilengkapi dengan fungsi `updateServo()` yang berfungsi mengatur waktu penutupan servo secara otomatis. Apabila waktu servo dalam kondisi terbuka telah mencapai nilai yang ditentukan pada `SERVO_OPEN_MS`, maka servo akan kembali ke posisi tertutup. Dengan demikian, mekanisme penutup tempat sampah dapat bekerja secara otomatis tanpa menggunakan fungsi `delay()` yang dapat menghentikan proses program lainnya.

Langkah 11 : Pemograman Tampilan Informasi pada LCD

```

167 // =====
168 // LCD
169 // =====
170 void cetakBarisBox(int baris, int sisaA, int sisaB) {
171     char buf[17];
172     snprintf(buf, sizeof(buf), "A:%3d% B:%3d%", sisaA, sisaB);
173     lcd.setCursor(0, baris);
174     lcd.print(buf);
175 }
176
177 void tampilIdle() {
178     int sisaA = hitungPersen(TRIG1, ECHO1);
179     int sisaB = hitungPersen(TRIG2, ECHO2);
180     char buf[17];
181
182     lcd.clear();
183     snprintf(buf, sizeof(buf), "L:%-3d  NL:%-3d", counterLogam, counterNonLogam);
184     lcd.setCursor(0, 0);
185     lcd.print(buf);
186
187     cetakBarisBox(1, sisaA, sisaB);
188 }

```

```

190 void tampilLogam() {
191     int sisaA = hitungPersen(TRIG1, ECHO1);
192     int sisaB = hitungPersen(TRIG2, ECHO2);
193     char buf[17];
194
195     lcd.clear();
196     snprintf(buf, sizeof(buf), "LOGAM      #%-3d", counterLogam);
197     lcd.setCursor(0, 0);
198     lcd.print(buf);
199
200     cetakBarisBox(1, sisaA, sisaB);
201 }
202
203 void tampilNonLogam() {
204     int sisaA = hitungPersen(TRIG1, ECHO1);
205     int sisaB = hitungPersen(TRIG2, ECHO2);
206     char buf[17];
207
208     lcd.clear();
209     snprintf(buf, sizeof(buf), "NON-LOGAM #%-3d", counterNonLogam);
210     lcd.setCursor(0, 0);
211     lcd.print(buf);
212
213     cetakBarisBox(1, sisaA, sisaB);
214 }
215
216 void tampilBoxPenuh(char box) {
217     int sisaA = hitungPersen(TRIG1, ECHO1);
218     int sisaB = hitungPersen(TRIG2, ECHO2);
219     char buf[17];
220
221     lcd.clear();
222     snprintf(buf, sizeof(buf), "!! BOX %c PENUH!", box);
223     lcd.setCursor(0, 0);
224     lcd.print(buf);
225
226     cetakBarisBox(1, sisaA, sisaB);
227 }

```

Gambar 4. 21 Pemograman yang ditampilkan pada LCD

Kode pada **Gambar 4.21** langkah ini berfungsi untuk mengatur informasi yang ditampilkan pada LCD I2C berdasarkan kondisi sistem. Fungsi cetakBarisBox() digunakan untuk menampilkan persentase sisa kapasitas pada masing-masing tempat sampah, yaitu tempat sampah logam dan nonlogam. Nilai

persentase diperoleh dari fungsi `hitungPersen()` berdasarkan pembacaan sensor ultrasonik.

Selanjutnya, fungsi `tampilIdle()` digunakan untuk menampilkan kondisi normal sistem beserta persentase kapasitas tempat sampah. Fungsi `tampilLogam()` menampilkan jumlah sampah logam yang telah terdeteksi, sedangkan `tampilNonLogam()` menampilkan jumlah sampah nonlogam. Sementara itu, fungsi `tampilBoxPenuh()` digunakan untuk memberikan informasi bahwa salah satu tempat sampah telah mencapai kondisi penuh. Dengan demikian, LCD berfungsi sebagai indikator informasi kondisi alat secara langsung kepada pengguna.

Langkah 12 : Pemograman Pengiriman Data Ke BLYNK

```

230 // DATA KE BLYNK
231 // =====
232 void cetakDataPV(float tegangan, float arus_A, float daya_W) {
233     Serial.println("-----");
234     Serial.println("PV");
235     Serial.print("  Tegangan : ");
236     Serial.print(tegangan, 2);
237     Serial.println(" V");
238     Serial.print("  Arus      : ");
239     Serial.print(arus_A, 2);
240     Serial.println(" A");
241     Serial.print("  Daya       : ");
242     Serial.print(daya_W, 2);
243     Serial.println(" W");
244     Serial.println("=====");
245 }
246
247 void kirimDataBlynk() {
248
249     // Data INA219 (monitoring PV)
250     float busVoltage  = ina219.getBusVoltage_V();
251     float shuntVoltage = ina219.getShuntVoltage_mV();
252     float current_mA   = ina219.getCurrent_mA();
253     float power_mW     = ina219.getPower_mW();
254     float loadVoltage  = busVoltage + (shuntVoltage / 1000.0);
255
256     Blynk.virtualWrite(V0, busVoltage);
257     Blynk.virtualWrite(V1, shuntVoltage);
258     Blynk.virtualWrite(V2, loadVoltage);
259     Blynk.virtualWrite(V3, current_mA);
260     Blynk.virtualWrite(V4, power_mW);
261
262     cetakDataPV(loadVoltage, current_mA / 1000.0, power_mW / 1000.0);
263
264     // Data sortir logam/non-logam
265     int sisaA = hitungPersen(TRIG1, ECHO1);
266     int sisaB = hitungPersen(TRIG2, ECHO2);
267
268     Blynk.virtualWrite(V5, counterLogam);
269     Blynk.virtualWrite(V6, counterNonLogam);
270     Blynk.virtualWrite(V7, sisaA);
271     Blynk.virtualWrite(V8, sisaB);
272 }

```

Gambar 4. 22 Pemograman pengiriman data ke BLYNK

Pada langkah ke-12 yang ditunjukkan oleh **Gambar 4.22** dilakukan pemrograman pengiriman data hasil pemantauan sistem ke platform Blynk IoT. Program melakukan pembacaan parameter kelistrikan menggunakan sensor

INA219 yang meliputi tegangan, arus, dan daya. Data hasil pembacaan tersebut selanjutnya dikirimkan ke Blynk melalui virtual pin V0 sampai V4, serta ditampilkan pada Serial Monitor sebagai informasi hasil pembacaan sensor. Selain parameter kelistrikan, program juga mengolah data persentase kapasitas tempat sampah logam dan nonlogam berdasarkan hasil pembacaan sensor ultrasonik. Data jumlah sampah logam dan nonlogam dikirim melalui virtual pin V5 dan V6, sedangkan data persentase kapasitas tempat sampah dikirim melalui virtual pin V7 dan V8. Dengan pemrograman tersebut, parameter kelistrikan dan kondisi kapasitas tempat sampah dapat dipantau melalui dashboard Blynk IoT secara real-time.

Langkah 13 : Inisialisasi Program Pada Fungsi setup()

```

274 // SETUP
275 // =====
276 void setup() {
277     Serial.begin(115200);
278     delay(1000);
279     Serial.println("=== TUGAS AKHIR ANANDYA ===");

```

Gambar 4. 23 Inisialisasi Program pada Fungsi setup()

Pada **Gambar 4.23** tahap ini dilakukan proses inisialisasi dan konfigurasi awal sistem melalui fungsi setup() yang dijalankan satu kali ketika ESP32 mulai aktif atau setelah proses *reset*. Program terlebih dahulu memuat data jumlah sampah logam dan nonlogam yang telah tersimpan sebelumnya menggunakan library Preferences, sehingga data penghitung dapat dipertahankan meskipun ESP32 mengalami *restart*. Selanjutnya, komunikasi I²C diinisialisasi menggunakan Wire.begin() untuk menghubungkan ESP32 dengan perangkat yang menggunakan komunikasi I²C, yaitu sensor INA219 dan LCD.

Program kemudian melakukan inisialisasi sensor INA219 dan menerapkan konfigurasi kalibrasi **32 V dan 2 A**. Sensor induktif dan sensor kapasitif dikonfigurasi sebagai input, sedangkan pin sensor ultrasonik diatur sesuai fungsi *trigger* dan *echo*. Dua motor servo juga diinisialisasi dan diarahkan ke posisi

tertutup sebagai kondisi awal sistem. Setelah itu, LCD diaktifkan dan digunakan untuk menampilkan informasi awal alat sebelum sistem menjalankan proses berikutnya.

Tahap selanjutnya adalah menghubungkan ESP32 ke jaringan Wi-Fi. Apabila koneksi berhasil, program melanjutkan proses koneksi ke platform Blynk untuk mendukung fungsi pemantauan sistem secara *real-time*. Apabila koneksi Wi-Fi gagal, sistem tetap dapat melanjutkan pengoperasian secara *offline*. Terakhir, BlynkTimer dikonfigurasi menggunakan interval BLYNK_UPDATE_MS untuk mengatur pengiriman data monitoring secara berkala. Setelah seluruh proses inisialisasi selesai, sistem menampilkan informasi bahwa alat telah siap digunakan.

```

373 // LOOP
374 // =====
375 void loop() {
376     Blynk.run();
377     timer.run();
378     updateServo();
379
380     bool adaLogam = bacaInduktif();
381     bool adaBenda = bacaKapasitif();
382
383     Serial.print("Induktif: ");
384     Serial.print(adaLogam ? "AKTIF" : "idle ");
385     Serial.print(" | Kapasitif: ");
386     Serial.println(adaBenda ? "AKTIF" : "idle ");
387
388     unsigned long sekarang = millis();

```

Gambar 4. 24 loop pemograman

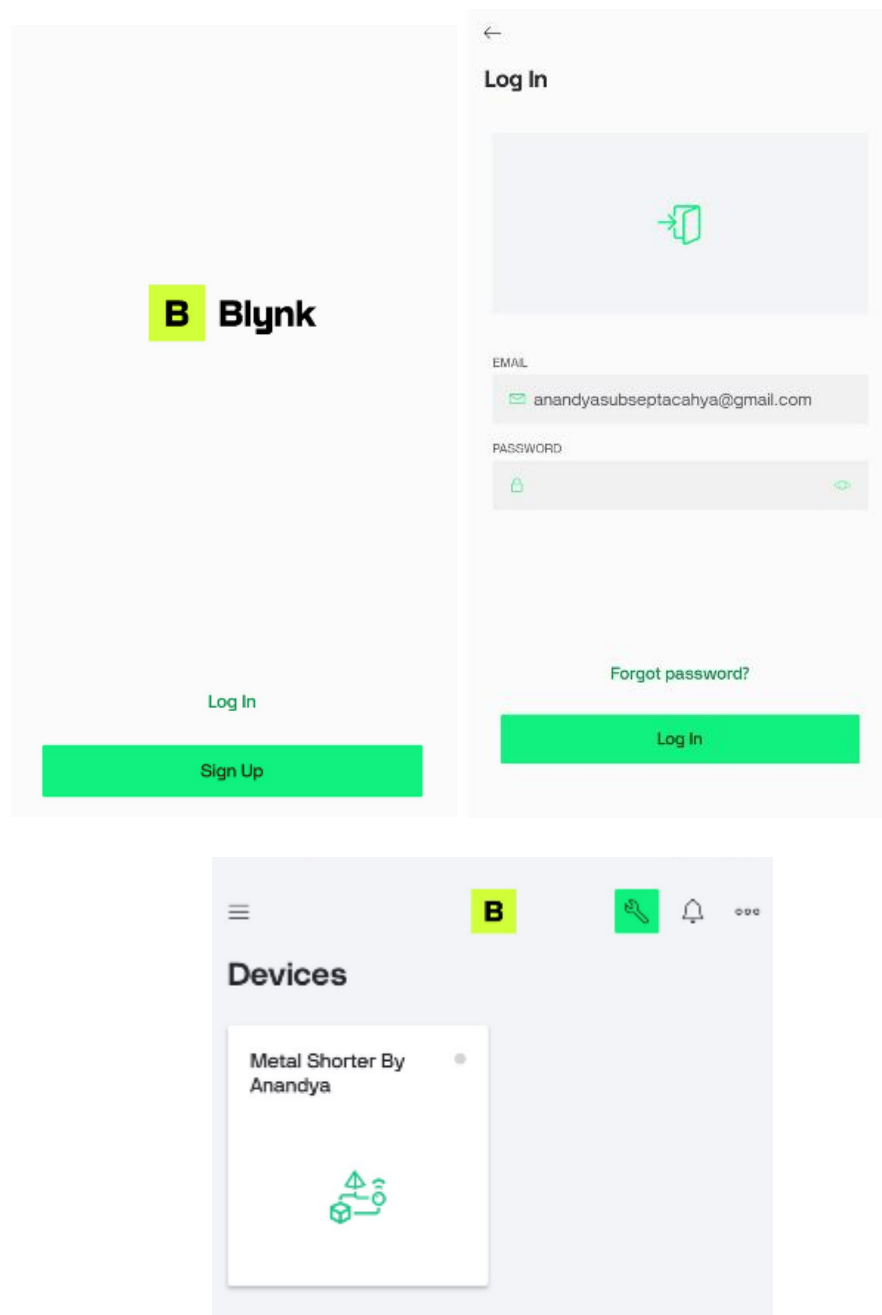
Setelah seluruh komponen dan komunikasi berhasil dikonfigurasi ditunjukkan pada **Gambar 4.24**, sistem berada dalam kondisi siap menjalankan proses utama pada fungsi loop().

4.2.2 Perangkat Lunak (*software*) Platform BLYNK

Blynk IoT merupakan platform Internet of Things (IoT) yang digunakan sebagai media komunikasi antara mikrokontroler dan perangkat pengguna melalui jaringan internet. Platform ini menyediakan layanan Blynk Cloud yang memungkinkan data dari mikrokontroler dikirim, disimpan, dan ditampilkan pada aplikasi Blynk secara real-time melalui jaringan Wi-Fi. Dengan adanya layanan tersebut, pengguna dapat memantau kondisi sistem dari jarak jauh menggunakan perangkat smartphone tanpa harus berada di lokasi alat. Blynk juga menyediakan berbagai widget yang dapat disesuaikan dengan kebutuhan sistem monitoring sehingga informasi yang diterima pengguna dapat ditampilkan dalam bentuk angka, grafik, maupun indikator visual.

Tahap awal penggunaan Blynk dilakukan dengan membuat template perangkat pada platform Blynk IoT. Template digunakan sebagai identitas proyek yang menghubungkan perangkat ESP32 dengan dashboard Blynk. Setelah template dibuat, diperoleh **Template ID**, **Template Name**, dan **Authentication Token** yang digunakan pada program ESP32 melalui konfigurasi `#define`. Ketiga parameter tersebut diperlukan agar ESP32 dapat melakukan autentikasi dan terhubung dengan server Blynk.

Langkah awal, dalam memanfaatkan platform Blynk dengan membuat akun dan melakukan login dengan akun email dan password.



Gambar 4. 25 Tampilan BYLNK

Pada Gambar 4.25 merupakan Tampilan Devices, Pencet pada platform Devices yang digunakan “Metal Shorter By Anandya” maka akan muncul tampilan pemantauan data Sampah Logam, Nonlogam dan kondisi Voltage, Arus, Watt pada panel surya yang dipantau menggunakan Sensor INA219. Kemudian terpantau juga pada Level Sampah dalam bentuk Persen (%) jika menunjukkan 0% maka

(KOSONG) jika 100% (PENUH). Fitur RESET DATA digunakan untuk mereset data yang sudah tersimpan dan bisa diatur kembali pada jumlah sampah Logam dan Nonlogam. Ditunjukkan pada **Gambar 4.26**



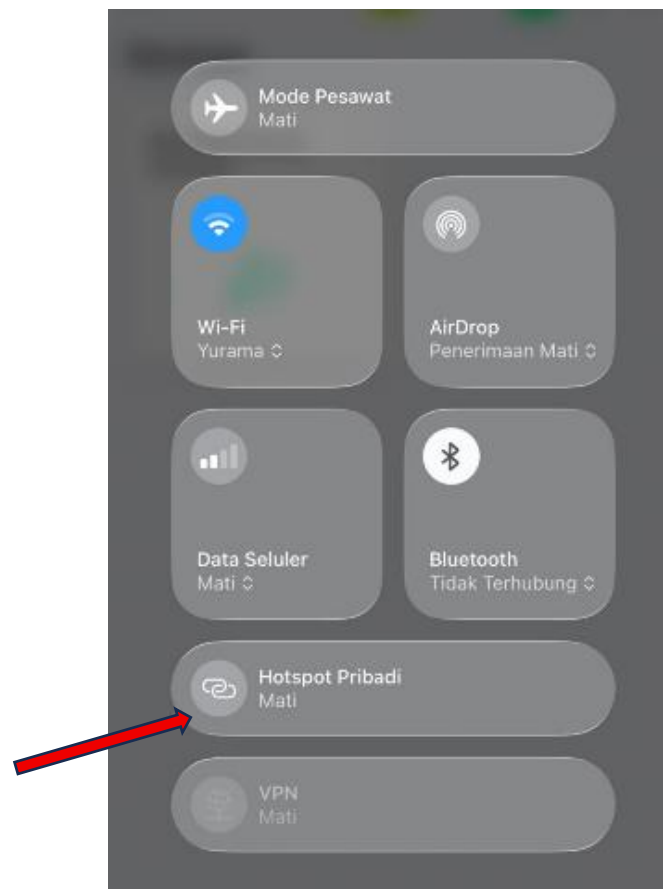
Gambar 4. 26 Tampilan Devices Blynk

Agar pemantauan data yang diperoleh secara realtime dapat di akses melalui smartphone dengan menyalakan data seluler dan menyambungkan Hostpot Pribadi, agar aplikasi Blynk tersambung dengan wifi smartphone maka kode pemograman

// **GANTI WIFI DISINI** pada char ssid[] dan char pass[] harus memasukkan id dan password yang sesuai pada Hostpot Pribadi. Ditunjukkan pada **Gambar 4.27**

```
13 // GANTI WIFI DI SINI
14 char ssid[] = "Anan";
15 char pass[] = "Anandya1809";
```

Gambar 4. 27 Pemograman untuk pergantian Wifi



Gambar 4. 28 Hotspot

Pada **Gambar 4.28** merupakan tampilan jika akan menghubungkan dengan Hotspot pribadi maka data dapat dipantau melalui smartphone masing masing.

4.3 Kalibrasi Sensor

Kalibrasi sensor merupakan proses untuk mengetahui dan menyesuaikan tingkat ketelitian hasil pembacaan sensor terhadap nilai atau kondisi acuan yang sebenarnya. Kalibrasi dilakukan untuk memastikan sensor yang digunakan pada sistem tong sampah pintar dapat memberikan hasil pembacaan yang sesuai sehingga data yang diperoleh dapat digunakan dengan baik dalam proses pengambilan keputusan sistem. Pada penelitian ini, kalibrasi dilakukan pada sensor proximity induktif, sensor proximity kapasitif, sensor HC-SR04, dan sensor INA219. Setiap sensor memiliki metode kalibrasi yang berbeda sesuai dengan karakteristik dan fungsi masing-masing sensor.

4.3.1 Kalibrasi Sensor Proximity Induktif LJ12A3-4-Z/BY

Kalibrasi sensor Proximity Induktif dilakukan untuk menentukan karakteristik pembacaan sensor untuk objek logam digunakan dalam sistem pemilahan sampah.

```

sketch_sep2a.ino
6  #define SENSOR_INDUKTIF 34
7
8  void setup() {
9
10     Serial.begin(115200);
11
12     pinMode(SENSOR_INDUKTIF, INPUT);
13
14     Serial.println();
15     Serial.println("=====");
16     Serial.println(" KALIBRASI SENSOR PROXIMITY INDUKTIF");
17     Serial.println("=====");
18     Serial.println();
19     Serial.println("1. Jauhkan semua benda LOGAM dari sensor.");
20     Serial.println("2. Amati nilai sensor.");
21     Serial.println("3. Dekatkan benda LOGAM ke sensor.");
22     Serial.println("4. Amati kembali nilai sensor.");

```

Output Serial Monitor X

Message (Enter to send message to 'ESP32 Dev Module' on 'COM11')

```

ADC Induktif: 0 | ADC Kapasitif: 769 | Induktif: idle | Kapasitif: idle
ADC Induktif: 0 | ADC Kapasitif: 771 | Induktif: idle | Kapasitif: idle
ADC Induktif: 0 | ADC Kapasitif: 767 | Induktif: idle | Kapasitif: idle
ADC Induktif: 0 | ADC Kapasitif: 767 | Induktif: idle | Kapasitif: idle
ADC Induktif: 0 | ADC Kapasitif: 769 | Induktif: idle | Kapasitif: idle
ADC Induktif: 0 | ADC Kapasitif: 771 | Induktif: idle | Kapasitif: idle
ADC Induktif: 0 | ADC Kapasitif: 765 | Induktif: idle | Kapasitif: idle

```

Gambar 4. 29 Kalibrasi sensor pada saat tidak ada objek logam

```

6  #define SENSOR_INDUKTIF 34
7
8  void setup() {
9
10     Serial.begin(115200);
11
12     pinMode(SENSOR_INDUKTIF, INPUT);
13
14     Serial.println();
15     Serial.println("=====");
16     Serial.println(" KALIBRASI SENSOR PROXIMITY INDUKTIF");
17     Serial.println("=====");
18     Serial.println();
19     Serial.println("1. Jauhkan semua benda LOGAM dari sensor.");
20     Serial.println("2. Amati nilai sensor.");
21     Serial.println("3. Dekatkan benda LOGAM ke sensor.");
22     Serial.println("4. Amati kembali nilai sensor.");

```

Output Serial Monitor X

Message (Enter to send message to 'ESP32 Dev Module' on 'COM11')

```

-----
ADC Induktif: 1876 | ADC Kapasitif: 2123 | Induktif: AKTIF | Kapasitif: AKTIF
ADC Induktif: 1873 | ADC Kapasitif: 2128 | Induktif: AKTIF | Kapasitif: AKTIF
ADC Induktif: 1882 | ADC Kapasitif: 2127 | Induktif: AKTIF | Kapasitif: AKTIF
ADC Induktif: 1885 | ADC Kapasitif: 2122 | Induktif: AKTIF | Kapasitif: AKTIF
ADC Induktif: 1882 | [LOGAM] Total: 3
Jarak: 23cm Isi:6% | [SPREADSHEET] WiFi tidak terhubung
ADC Induktif: 1881 | ADC Kapasitif: 2128 | Induktif: AKTIF | Kapasitif: AKTIF

```

Gambar 4. 30 Kalibrasi Sensor saat mendeteksi objek logam

Dengan mengamati nilai pembacaan sensor pada kondisi tanpa objek ditunjukkan pada **Gambar 4.29** dan pada saat diberikan objek logam dengan jarak tertentu ditunjukkan pada **Gambar 4.30**. Nilai pembacaan sensor diamati pada nilai ADC ESP32 pada GPIO34 dan ditampilkan di Serial Monitor.

4.3.2 Kalibrasi Sensor Proximity Kapasitif LJC18A3-H-Z/BY

Proses Kalibrasi sensor dilakukan dengan mengamati nilai pembacaan sensor pada kondisi tanpa objek dan saat diberikan objek nonlogam.

```

7
8 void setup() {
9
10     Serial.begin(115200);
11
12     pinMode(SENSOR_KAPASITIF, INPUT);
13
14     analogReadResolution(12);
15
16     Serial.println();
17     Serial.println("=====");
18     Serial.println(" KALIBRASI SENSOR PROXIMITY KAPASITIF");
19     Serial.println("=====");
20     Serial.println();
21     Serial.println("1. Jauhkan benda dari sensor.");
22     Serial.println("2. Amati nilai sensor.");
23     Serial.println("3. Dekatkan benda NON-LOGAM.");
24     Serial.println("4. Amati kembali nilai sensor.");
25     Serial.println();
26 }

```

Output Serial Monitor X

Message (Enter to send message to 'ESP32 Dev Module' on 'C... New Line 115200

Nilai Sensor Kapasitif = 781
 Nilai Sensor Kapasitif = 782
 Nilai Sensor Kapasitif = 784
 Nilai Sensor Kapasitif = 784
 Nilai Sensor Kapasitif = 784
 Nilai Sensor Kapasitif = 781
 Nilai Sensor Kapasitif = 784

Gambar 4. 31 Kalibrasi sensor kapasitif saat tidak mendeteksi objek nonlogam

```

7
8 void setup() {
9
10     Serial.begin(115200);
11
12     pinMode(SENSOR_KAPASITIF, INPUT);
13
14     analogReadResolution(12);
15
16     Serial.println();
17     Serial.println("=====");
18     Serial.println(" KALIBRASI SENSOR PROXIMITY KAPASITIF");
19     Serial.println("=====");
20     Serial.println();
21     Serial.println("1. Jauhkan benda dari sensor.");
22     Serial.println("2. Amati nilai sensor.");
23     Serial.println("3. Dekatkan benda NON-LOGAM.");
24     Serial.println("4. Amati kembali nilai sensor.");
25     Serial.println();
26 }

```

Output Serial Monitor X

Message (Enter to send message to 'ESP32 Dev Module' on 'C... New Line 115200

Nilai Sensor Kapasitif = 2148
 Nilai Sensor Kapasitif = 2145
 Nilai Sensor Kapasitif = 2144
 Nilai Sensor Kapasitif = 2143
 Nilai Sensor Kapasitif = 2143
 Nilai Sensor Kapasitif = 2145
 Nilai Sensor Kapasitif = 2146

Gambar 4. 32 Kalibrasi sensor kapasitif saat mendeteksi objek nonlogam

Pada **Gambar 4.31** dan **Gambar 4.32** merupakan Pembacaan sensor dilakukan melalui ADC ESP32 pada GPIO35 dan ditampilkan melalui Serial Monitor. Berdasarkan hasil tersebut ditentukan nilai *threshold* yang digunakan pada

program untuk menentukan kondisi sensor pada saat diberikan objek yang terdeteksi dan tidak terdeteksi.

4.3.3 Kalibrasi Sensor HC-SR04

Kalibrasi sensor HC-SR04 dilakukan untuk mengetahui ketelitian sensor dalam mengukur jarak yang digunakan sebagai parameter tingkat kepenuhan tempat sampah. Pengujian dilakukan dengan memberikan beberapa variasi jarak objek terhadap sensor. Jarak sebenarnya diukur menggunakan penggaris atau meteran sebagai alat ukur pembanding, kemudian dibandingkan dengan hasil pembacaan HC-SR04 yang ditampilkan melalui Serial Monitor.



Gambar 4. 33 Kalibrasi Sensor HC-SR04 Bak Logam

Pada **Gambar 4.33** dan **Gambar 4.34** merupakan Kalibrasi sensor HC-SR04 pada Bak Logam dan Nonlogam dilakukan dengan dua cara menggunakan penggaris dan melihat tampilan pada Serial Monitor, kedua hasil tersebut kemudian diperhitungkan untuk mendapatkan nilai error dan akurasi sensor sensor HC0SR04 pada Bak Logam berdasarkan **persamaan 2.1** untuk nilai adalah 3,85% dan berdasarkan **persamaan 2.2** untuk nilai akurasi adalah 96,15%.



Gambar 4. 34 Kalibrasi sensor HC-SR04 pada Bak NonLogam

Berdasarkan nilai kalibrasi sensor HC-SR04 yang diperoleh pada bak Non Logam maka untuk mendapatkan nilai error dan akurasi sensor sensor HC0SR04 pada Bak Non Logam berdasarkan **persamaan 2.1** untuk nilai adalah 4% dan berdasarkan **persamaan 2.2** untuk nilai akurasi adalah 96%.

4.3.4 Kalibrasi INA219

Kalibrasi sensor INA219 dilakukan untuk mengetahui ketelitian sensor dalam mengukur parameter kelistrikan pada sistem catu daya. Parameter yang dibandingkan meliputi tegangan, daya, dan arus. Multimeter digital digunakan sebagai alat ukur pembanding, sedangkan hasil pembacaan INA219 diperoleh melalui program pada ESP32. Ditunjukkan pada **Gambar 4.35** dan **Gambar 4.36**

```

9 // SDA = GPIO 4, SCL = GPIO 5
10 Wire.begin(4, 5);
11
12
13 variable ina219 terdeteksi!";
14 Type: class Adafruit_INA219
15 Adafruit_INA219 ina219
16 ina219.setCalibration_32V_2A();
17
18 Serial.println("=== KALIBRASI INA219 ===");
19 }
20
21 void loop() {
22 float busVoltage = ina219.getBusVoltage_V();
23 float shuntVoltage = ina219.getShuntVoltage_mV();
24 float current_mA = ina219.getCurrent_mA();
25 float power_mW = ina219.getPower_mW();
26
27

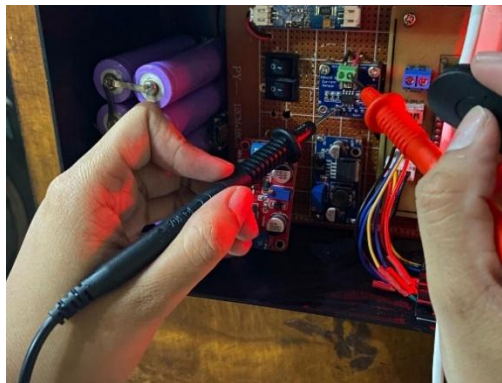
```

Output Serial Monitor X

Message (Enter to send message to 'ESP32 Dev Module' on 'C... New Line

Power : 0.00 mW

Bus Voltage : 1.212 V
Shunt Voltage : 0.000 mV
Load Voltage : 1.212 V
Current : 0.00 mA
Power : 0.00 mW



Gambar 4. 35 Kalibrasi Sensor INA219 parameter Tegangan

Kemudian hasil kedua pengukuran dengan parameter tegangan tersebut pada sensor INA219 diperhitungkan dengan perhitungan berdasarkan persamaan 2.1 untuk nilai error maka nilai error adalah 2,18% dan untuk nilai akurasi berdasarkan persamaan 2.2 maka nilai akurasinya adalah 97,82%. Pada pengujian tegangan, multimeter digunakan untuk mengukur tegangan pada titik yang sama dengan tegangan yang dipantau oleh INA219.

Sementara itu, pengukuran arus dilakukan dengan menempatkan alat ukur secara seri pada jalur beban. Hasil pembacaan multimeter dan INA219 kemudian dicatat dan dibandingkan.

```

-----
PV
Tegangan : 8.79 V
Arus      : 4.60 mA
Daya      : 40.00 mW
=====

```

Metal Shorter By Anandya •		
Voltage	Arus	Watt
8,81 ^v	4,1 ^{mA}	40 ^{mW}



Gambar 4. 36 Kalibrasi Sensor INA219 parameter Arus

Kemudian hasil kedua pengukuran dengan parameter arus tersebut diperhitungkan dengan persamaan 2.1 untuk nilai error adalah 0,66% dan persamaan 2.2 untuk nilai akurasi nya adalah 99,34%